3
Algoritmo NCE

Neste capítulo descrevemos o algoritmo `NCE` (*News Content Extractor*), desenvolvido para extrair o conteúdo relevante de uma página de notícia de uma forma simples e eficiente.

O `NCE` é dividido em duas etapas: a busca pelo nó separador e o refinamento da subárvore do nó separador. Na Seção 3.1 explicamos a hipótese do nó separador, na qual se baseia o algoritmo `NCE`, e também como o processo de busca pelo nó separador é efetuado. Em seguida na Seção 3.2 descrevemos os refinamentos no algoritmo `NCE`. Finalmente, na seção 3.3 fazemos uma análise do algoritmo `NCE`.

PUC-Rio - Certificação Digital Nº 0711277/CA

3.1
Busca do nó separador

A seguir introduzimos algumas notações que são importantes para tornar a explicação da hipótese do nó separador mais simples e clara.

Seja um nó u pertencente a uma árvore DOM T, a subárvore enraizada em u que inclui u e todos seus descendentes, é denotada por T_u. Um conjunto de árvores F é chamado de floresta. Dada uma floresta F, $texto(F)$ representa o texto localizado nos nós em F. A densidade de *links* de uma floresta F, representada por $densidadeLink(F)$, é a razão entre o número de caracteres contidos em *tags* do tipo `<a>` em F e $|texto(F)|$, onde $|texto(F)|$ é o número de caracteres que $texto(F)$ possui. A relação $pai(u)$ significa que u é filho de $pai(u)$. A operação de subtração entre duas subárvores T_t e T_u, representada por $T_t - T_u$, é a interseção dos nós de T_t e T_u.

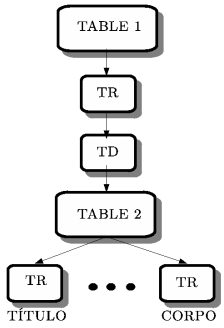
Utilizamos também o conceito de uma função $f_B : S \mapsto \mathbb{N}$ que associa cada elemento de S com sua multiplicidade em B, chamamos esta função de *bag* B. Usamos $|B|$ para denotar o número de elementos (não necessariamente distintos) em B. Um exemplo de *bag* pode ser representada pela tupla $B = (bola, gato, gato, tia)$ então $|B| = 4$.

Dado um número real positivo tr, utilizamos $F_{u,tr}$ para representar o conjunto de árvores com densidade de *link* de no máximo tr que são enraizadas nos filhos de u.

A hipótese do nó separador pode ser enunciada da seguinte forma:

Hipótese 1 (Existência do nó separador.) *Para uma página de notícia P, existe um nó u na árvore DOM associada a P e um número real positivo tr para o qual a união dos blocos de textos da floresta $F_{u,tr}$ é uma boa aproximação para o conteúdo relevante de P.*

3.1(a): Exemplo de uma página web



3.1(b): Esboço de árvore DOM da página da Figura 3.1(a) e nó separador

Veja na Figura 3.1(a) uma página de notícias. Um esboço da sua árvore DOM pode ser vista na Figura 3.1(b). O nó separador, neste caso, é o nó da *tag TABLE 2*. Neste exemplo o nó separador não contém nenhum excesso e não perde conteúdo, porém em alguns casos não existe um nó separador que produza um resultado tão preciso.

A busca do nó separador é feita em duas etapas, a primeira realiza o parser da página e a segunda realiza uma busca em profundidade na árvore seguindo um conjunto de regras. Segue uma descrição mais detalhada destes dois passos.

Na execução do parser da página é construída uma árvore com a mesma topologia de uma árvore DOM, porém com o atributo adicional que representa o identificador do nó atual u que significa a quantidade de nós visitados antes do nó u em uma busca em profundidade.

Após o parser, a árvore DOM é percorrida por uma busca em profundidade (DFS). Quando a busca chega em uma folha é verificado se esta folha possui um número mínimo de caracteres α_0. Se isto ocorre o procedimento `SelectAncestor` é executado. Este percorre os ancestrais da folha em ordem

PUC-Rio - Certificação Digital Nº 0711277/CA

crescente de distância, onde distância entre dois nós é o número de ramos que há no caminho entre eles. O procedimento `SelectAncestor` pára no nó ancestral t que satisfaz simultaneamente as quatro condições abaixo:

1. O nó t não é parágrafo;
2. T_t deve ter no mínimo α_1 caracteres;
3. $densidadeLink(pai(T_t) - T_t) \geq \alpha_2$ e $|texto(pai(T_t) - T_t)| \geq \alpha_3$;
4. t tem no máximo um irmão t', cujo número de caracteres é maior que α_2 e cuja densidade de *links* é menor que α_3.

Se nenhum nó na árvore satisfaz estas condições a árvore toda é retornada como resposta, caso contrário ele retorna $F_{t,tr}$, onde $tr = 0.4$. O valor de tr foi obtido a partir de testes com o corpus de exploração. Estes testes foram feitos com alguns valores entre zero e um até obter um nó separador satisfatório para cada página do corpus de exploração.

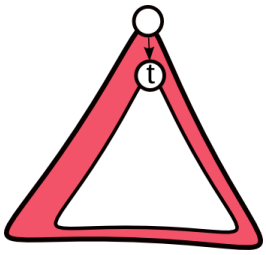


Figura 3.1: Região que pode ser acrescentada como conteúdo relevante

O item 2 assegura um bloco grande o suficiente para ser o conteúdo relevante da notícia.

O item 3 da lista avalia se vale a pena substituir o nó atual t por seu pai. Veja o esboço de uma árvore DOM na Figura 3.1. Através desta regra o algoritmo julga se a região colorida acrescenta conteúdo não relevante. Este julgamento é feito com base na densidade de *links* e na quantidade de palavras, pois se a região possuir uma quantidade de texto razoável e a densidade de *links* for alta é muito provável que a região inclua conteúdo não relevante. Veja que uma região com densidade de *link* alta não significa necessariamente conteúdo não relevante. Uma notícia pode ter um parágrafo que é um bloco de texto com um *link* que referência outro conteúdo, como por exemplo algumas fotos. Este parágrafo pode apresentar uma densidade de *links* alta devido a proporção do tamanho do texto e do *link*, porém isso só acontece devido ao pequeno tamanho do texto. A densidade de *links* é limitada pelo parâmetro α_2 e a quantidade de caracteres é limitada pelo parâmetro α_3.

PUC-Rio - Certificação Digital Nº 0711277/CA

O item 4 tenta evitar que o método perca conteúdo relevante impedindo a busca de parar no nó t se ele possuir pelo menos dois irmãos com certa quantidade de texto.

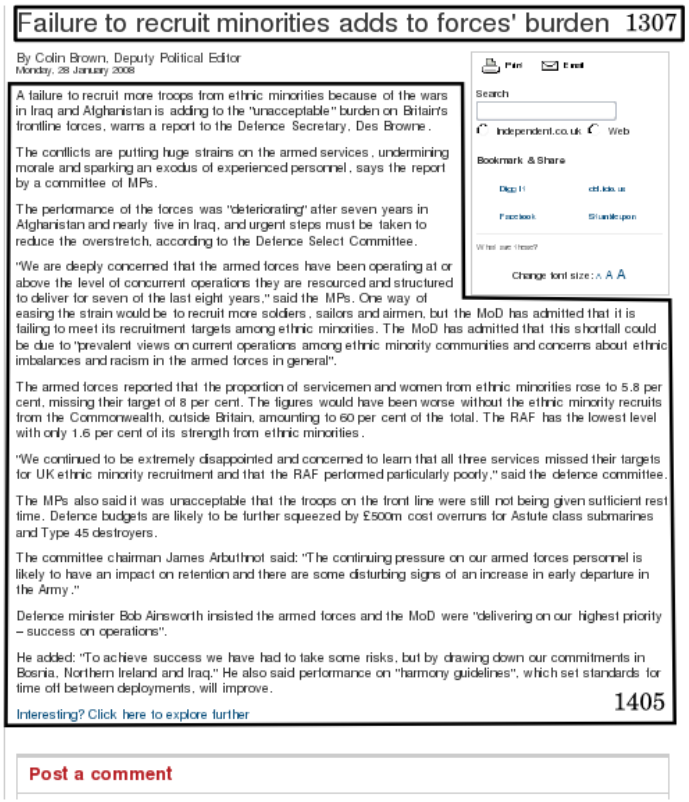


Figura 3.2: Corpo e título de uma notícia

Por exemplo, seja o corpo de uma notícia apresentado na Figura 3.2 e os comentários desta notícia na Figura 3.3. O esboço da árvore DOM desta notícia é exibido pela Figura 3.4.

Os números nas Figuras 3.2, 3.3 e 3.4 representam identificadores de nós. O identificador de um retângulo representa o identificador da raiz da subárvore que renderiza aquele retângulo. Por exemplo, o comentário cujo rótulo é 1590 é composto por uma subárvore com os nós 1592 a 1603. Este conjunto de nós ao ser renderizado pelo navegador mostra o retângulo com o rótulo 1590. O nó 1299 engloba a Figura 3.2 e 3.3, porém para uma resolução melhor da região da notícia e dos comentários ele não foi destacado.

O conteúdo retornado na primeira etapa do algoritmo `NCE` é a floresta $F_{1299,0.4}$. A densidade de *links* 0.4 foi escolhida após testes no corpus de exploração. Neste exemplo, só uma subárvore é retornada, porém em alguns casos outros irmãos da subárvore detectada podem ser retornados. O resultado retornado, neste caso, captura os comentários, considerados conteúdo irrelevante.

PUC-Rio - Certificação Digital Nº 0711277/CA

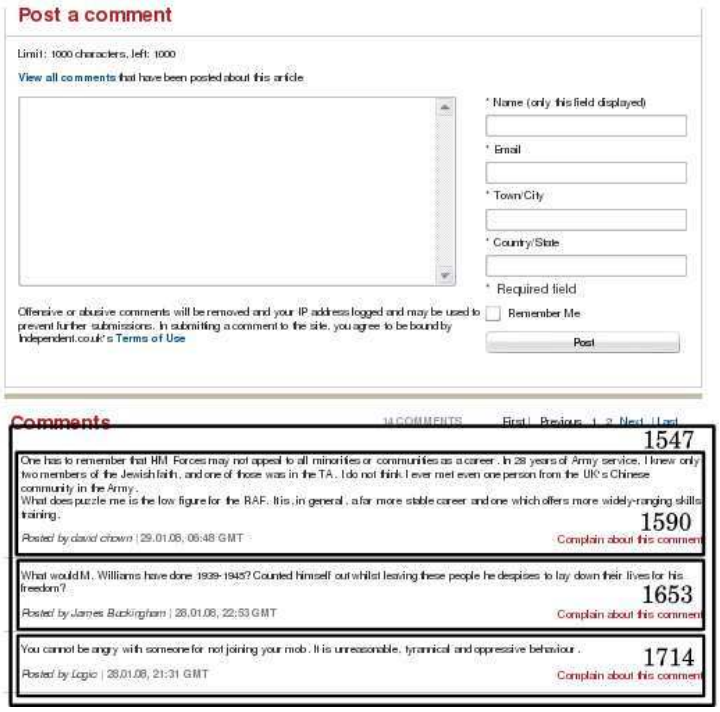


Figura 3.3: Comentários da notícia

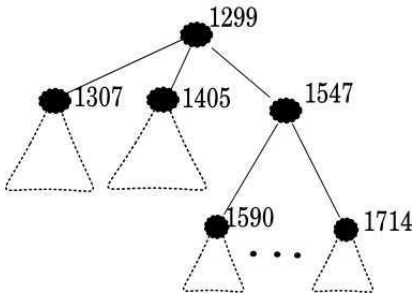


Figura 3.4: Esquema da árvore DOM do documento das Figuras 3.2 e 3.3

3.2 Refinando a solução

Com o objetivo de refinar a solução retornada pelo nó separador, duas etapas foram acrescentadas: a remoção de comentários e a busca do título. Os comentários muitas vezes introduzem um excesso de conteúdo grande e isso motivou o aperfeiçoamento da busca do nó separador. O título é uma perda quantitativa pequena, porém seu papel é importante na notícia, por esta razão o acréscimo da etapa da busca por título.

A etapa de remoção de comentários é baseada na seguinte hipótese.

Hipótese 2 *Comentários estão localizados depois do corpo da notícia em uma região que tem padrões repetitivos, como por exemplo data e autor.*

PUC-Rio - Certificação Digital Nº 0711277/CA

Veja na Figura 3.3 no rodapé de cada bloco de comentário, ou a subárvore onde o comentário está contido, que existe uma referência ao autor e a data do comentário. Assim nesta fase o algoritmo procura por blocos que apresentam conteúdo similar na floresta U retornada como solução na fase da busca do nó separador. Se existirem tais blocos, então todo o conteúdo que aparece a partir do primeiro bloco do conjunto de blocos é descartado porque é muito provável que tal bloco inicie a região de comentários.

O processo de remoção começa com uma DFS na floresta U. Cada nó $v \in U$, cujo conteúdo possuir um número de caracteres em um intervalo dado $[\alpha_4, \alpha_5]$, é inserido em uma lista L e os descendentes de v são ignorados pela DFS, visto que o nó v contém o texto de seus descendentes.

O passo seguinte é construir um grafo $G = (L, E)$. As arestas $(u, v) \in E$ devem seguir as seguintes restrições:

1. Existem no máximo α_6 elementos entre u e v em L;

2. O tamanho da maior subsequência comum entre $texto(T_u)$ e $texto(T_v)$ é pelo menos α_7 vezes o tamanho do menor destes dois textos, onde α_7 é um número entre zero e um.

   Os parâmetros α_6 e α_7 são dados como entrada.

Logo que a primeira componente conexa C em G, com no mínimo 3 nós, por exemplo n_1, n_2 e n_3, é construída, o primeiro ancestral n em comum dos dois primeiros nós n_1 e n_2 em C é localizado. A subárvore enraizada no filho de n que contém n_1 é descartada, assim como os nós que aparecem depois dele em U.

A motivação ao construir este grafo é relacionar nós em um determinado nível de semelhança e proximidade, pois observe que uma componente conexa em G corresponde a um conjunto de subárvores em U que são similares e próximas.

No exemplo da Figura 3.4 o módulo de remoção de comentários identifica similaridade entre as subárvores a partir do nó 1590, pois o cabeçalho contendo autor e data do comentário ocorrem em todos eles. O nó identificado como ancestral em comum das subárvores é o nó 1547.

As raízes dos nós com cabeçalho acabam se relacionando no grafo G por serem próximas uma das outras. A proximidade é fácil identificar visualmente e também pelos identificadores das raízes dos nós. Por esta razão os nós a partir do 1590 são excluídos da solução inicial.

Outro módulo que refina a solução é a procura do título. Este módulo se baseia nos seguintes fatos observados em diversas páginas de notícias: o título é reproduzido na notícia na *tag* `<title>` da página; o título da notícia

PUC-Rio - Certificação Digital Nº 0711277/CA

se localiza antes do corpo da notícia em uma DFS; o título é constituído de poucas palavras e algumas palavras do título aparecem no corpo da notícia.

Inicialmente a busca do título visita os nós $v \in T$ e cujo identificador é menor que o identificador do primeiro nó de U', que é a floresta retornada pelo módulo de remoção de comentários. Nesta etapa os nós que satisfazem as condições a seguir são inclusos em uma lista L':

1. Se v é associado com a *tag* `<title>`;
2. Se $|bag(text(T_v))| \leq \alpha_8$ e $|bag(text(T_v)) \cap bag(text(U'))| \geq \alpha_9|bag(text(T_v))|$.

O primeiro item destas condições é motivado por uma inspeção no corpus, que mostrou que esta *tag* muitas vezes contém o título da notícia. O segundo item analisa os nós que contém no máximo um número determinado de palavras e o quanto do texto deste nó tem em comum com o texto da floresta retornada pelo módulo anterior, que assumimos conter o conteúdo da página.

Ao final deste processo o nó em L' que possuir o maior texto é incluso na solução do `NCE`.

Através da inspeção do corpus de exploração (CE), os parâmetros α_0, α_1, α_2, α_3, α_4, α_5, α_6, α_7, α_8 e α_9 são inicializados. Os valores padrão são valores iniciais que os parâmetros recebem e que foram escolhidos a partir da inspeção manual do corpus CE e variados dentro de uma vizinhança. Na Tabela 3.1 cada parâmetro é listado com seu respectivo valor padrão, que é o valor afixado para os testes.

3.3 Complexidade

Seja a entrada do algoritmo um árvore DOM T. O número de nós nesta árvore chamamos de l.

Na primeira fase do `NCE`, onde é realizada a busca do nó separador, uma DFS é executada na árvore T até encontrar uma folha com pelo menos α_0 caracteres. Este procedimento tem complexidade $O(l)$. Durante esta busca em profundidade cada nó é verificado se é uma *tag* `<p>` e o limite inferior do tamanho do texto do nó é pelo menos α_1. Estas duas condições são realizadas em passo constante, visto que estes atributos foram computados durante a construção da árvore. Outra condição que é testada durante a DFS é a densidade de *links* e o limite superior de caracteres do irmão do nó corrente, este passo é linear. Portanto a complexidade desta fase se mantém em $O(l)$.

A segunda fase do algoritmo é a extração de comentários. Nela é realizada uma DFS a fim de construir uma lista de candidatos a comentários, chamada

PUC-Rio - Certificação Digital Nº 0711277/CA

Tabela 3.1: Valores padrão dos parâmetros do algoritmo `NCE`

Parâmetro	Descrição	Valor
α_0	Número mínimo de caracteres para folhas.	200
α_1	Número mínimo de caracteres na subárvore analisada.	600
α_2	Número mínimo de palavras em relação ao pai.	0.1
α_3	Densidade mínima de *links*.	0
α_4	Número mínimo de caracteres do nó para montar o grafo de remoção de comentários.	30
α_5	Número máximo de caracteres do nó para montar o grafo de remoção de comentários.	120
α_6	Distância máxima entre dois nós para ser inserido grafo utilizado para remover comentários.	5
α_7	A razão mínima que o texto entre dois nós deve ter para ser inserido no grafo de remoção de comentários.	0.8
α_8	Número mínimo de palavras que um candidato a título deve ter.	4
α_9	Taxa de similaridade entre candidatos a títulos.	0.7

na Seção 2 de L. Os nós com tamanho de texto entre α_4 e α_5 são inseridos nesta lista. A DFS neste passo tem complexidade $O(l_u)$, onde l_u é o número de nós da floresta U, retornada como resposta pela fase da busca do nó separador.

Cada nó incluído na lista L é testado com os nós cuja posição em L é distante no máximo α_6 posições. Este teste consiste em comparar o texto dos nós utilizando o algoritmo de Maior Subsequência em Comum (*Longest Common Subsequence*) para verificar se um nó tem algo em comum com os nós cuja posição em L é no máximo α_6.

A implementação utilizada no `NCE` para o cálculo da maior subsequência em comum entre duas strings A e B foi proposta por Kuo e Cross em (KC89). Este algoritmo possui a complexidade:

$$O(r + n(LLCS(A, B) + \log n)) \tag{3-1}$$

,onde r é o tamanho do conjunto $\{(i, j)|A[i] = B[j]\}$, $LLCS(A, B)$ é o tamanho da maior subsequência em comum entre as strings A e B e n é o tamanho da string A mais o tamanho da string B.

Exeprimentos realizados por Kuo e Cross mostram que para grandes valores de n o tempo cresce quase linearmente. Assim, na prática, o algoritmo não tem impacto quadrático no `NCE`. O algoritmo LCS é executado em $\Theta(\alpha_6|L|)$

PUC-Rio - Certificação Digital Nº 0711277/CA

vezes.

A fase de recuperação dos títulos constrói uma lista L' iniciando um DFS na árvore T até encontrar o primeiro nó pertencente a U', que é a floresta retornada pelo módulo anterior. Esta DFS possui a complexidade $O(l)$. Durante esta construção existe a comparação dos nós cujo tamanho é até α_8 com os nós da floresta U'. Esta comparação é feita através de saco de palavras, que é linear no número de palavras que a floresta U' contém.

PUC-Rio - Certificação Digital Nº 0711277/CA